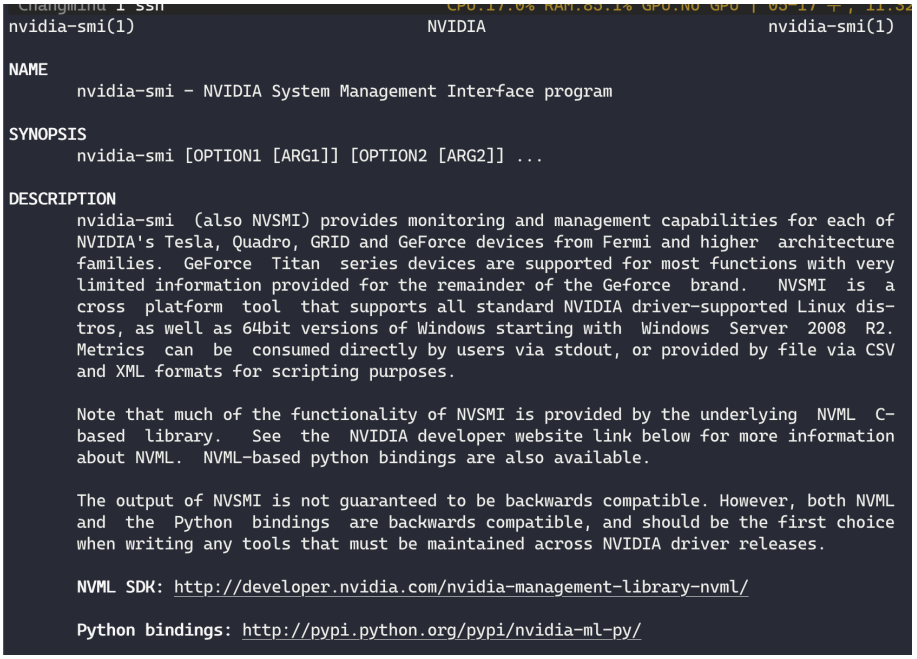
HPC: HW4 Report

2017-19841 최창민

1. GPU정보 확인하기

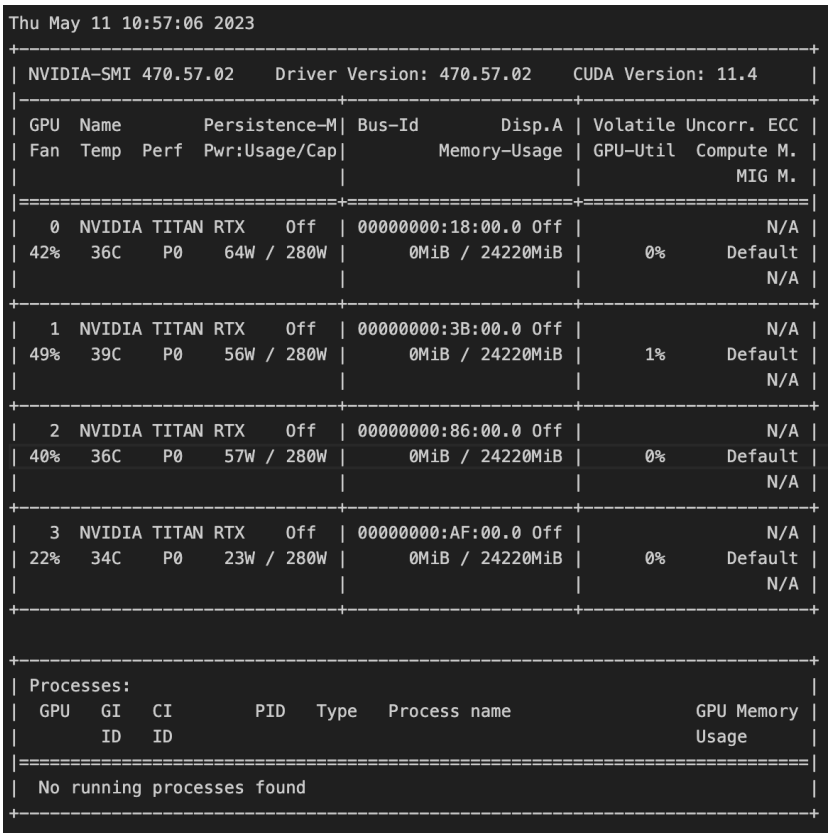
(a) 커맨드의 의미와 실행 결과

- srun --partition=shpc --gres=gpu:4
    - shpc 파티션에서 노드 1개의 gpu 4개를 할당
- srun --partition=shpc --gres=gpu:4 nvidia-smi
    - man nvidia-smi를 통해 커맨드의 의미를 알아봤습니다.



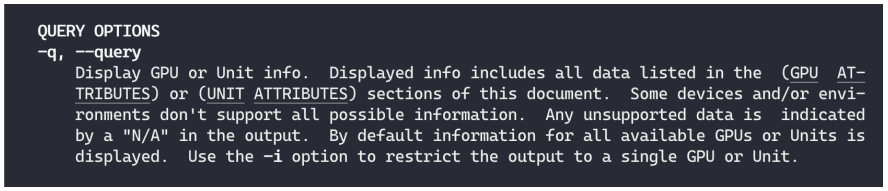
```
nvidia-smi(1)                          NVIDIA                          nvidia-smi(1)

NAME
       nvidia-smi - NVIDIA System Management Interface program

SYNOPSIS
       nvidia-smi [OPTION1 [ARG1]] [OPTION2 [ARG2]] ...

DESCRIPTION
       nvidia-smi  (also NVSMI) provides monitoring and management capabilities for each of
       NVIDIA's Tesla, Quadro, GRID and GeForce devices from Fermi and higher  architecture
       families.  GeForce  Titan  series devices are supported for most functions with very
       limited information provided for the remainder of the Geforce  brand.  NVSMI  is  a
       cross  platform  tool  that supports all standard NVIDIA driver-supported Linux dis-
       tros, as well as 64bit versions of Windows starting with  Windows  Server  2008  R2.
       Metrics  can  be  consumed directly by users via stdout, or provided by file via CSV
       and XML formats for scripting purposes.

       Note that much of the functionality of NVSMI is provided by the underlying  NVML  C-
       based  library.   See  the  NVIDIA developer website link below for more information
       about NVML.  NVML-based python bindings are also available.

       The output of NVSMI is not guaranteed to be backwards compatible. However, both NVML
       and  the  Python  bindings  are backwards compatible, and should be the first choice
       when writing any tools that must be maintained across NVIDIA driver releases.

       NVML SDK: http://developer.nvidia.com/nvidia-management-library-nvml/

       Python bindings: http://pypi.python.org/pypi/nvidia-ml-py/
```

- 실행 결과



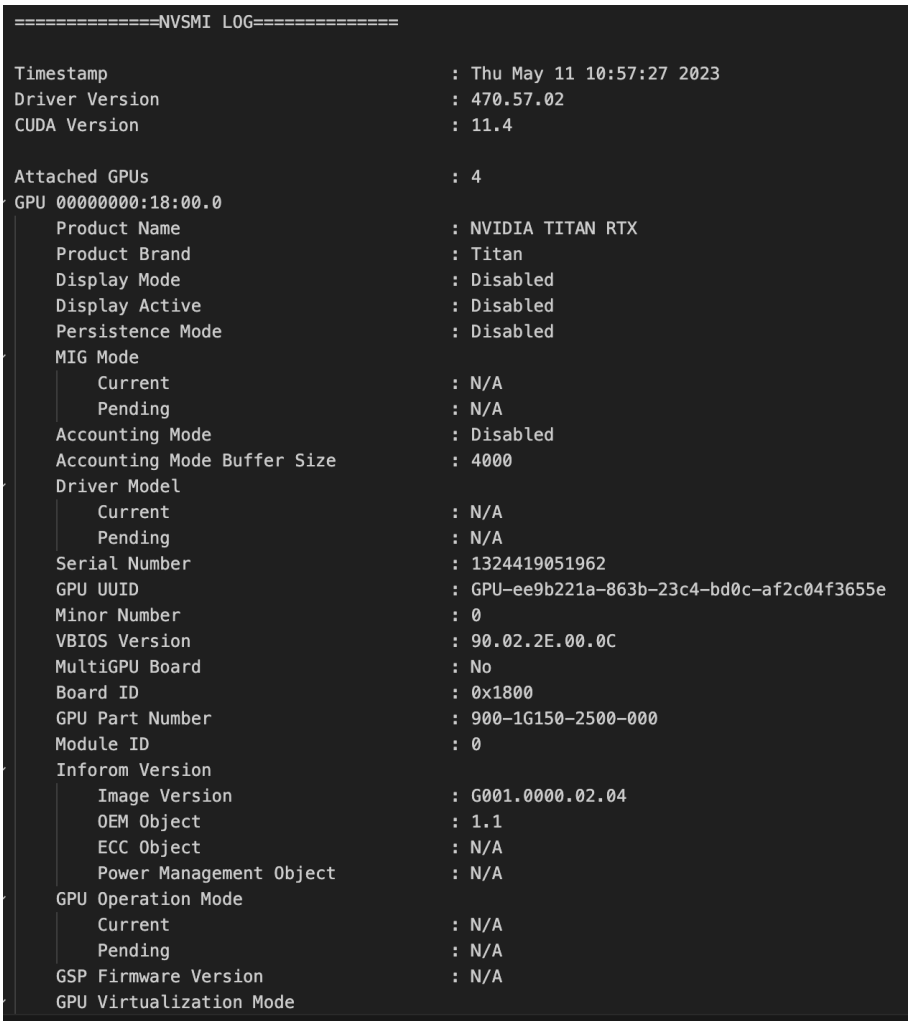
```
Thu May 11 10:57:06 2023
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 470.57.02    Driver Version: 470.57.02    CUDA Version: 11.4     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  NVIDIA TITAN RTX    Off  | 00000000:18:00.0 Off |                  N/A |
| 42%   36C    P0    64W / 280W |      0MiB / 24220MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  NVIDIA TITAN RTX    Off  | 00000000:3B:00.0 Off |                  N/A |
| 49%   39C    P0    56W / 280W |      0MiB / 24220MiB |      1%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  NVIDIA TITAN RTX    Off  | 00000000:86:00.0 Off |                  N/A |
| 40%   36C    P0    57W / 280W |      0MiB / 24220MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  NVIDIA TITAN RTX    Off  | 00000000:AF:00.0 Off |                  N/A |
| 22%   34C    P0    23W / 280W |      0MiB / 24220MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

- srun --partition=shpc --gres=gpu:4 nvidia-smi -q
    - 마찬가지로 man nvidia-smi에서 해당 옵션의 의미를 알아봤습니다.



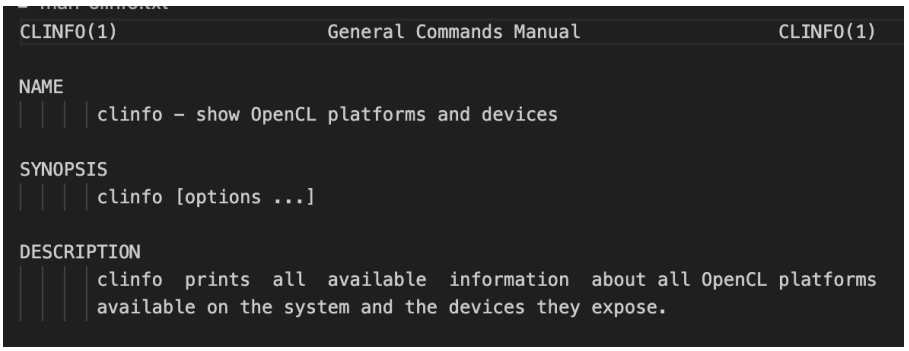
```
QUERY OPTIONS
-q, --query
    Display GPU or Unit info.  Displayed info includes all data listed in the  (GPU  ATTRIBUTES) or (UNIT ATTRIBUTES) sections of this document.  Some devices and/or environments don't support all possible information.  Any unsupported data is  indicated by a "N/A" in the output.  By default information for all available GPUs or Units is displayed.  Use the -i option to restrict the output to a single GPU or Unit.
```

- 실행 결과



```
==============NVSMI LOG==============

Timestamp                                 : Thu May 11 10:57:27 2023
Driver Version                            : 470.57.02
CUDA Version                              : 11.4

Attached GPUs                             : 4
GPU 00000000:18:00.0
    Product Name                          : NVIDIA TITAN RTX
    Product Brand                         : Titan
    Display Mode                          : Disabled
    Display Active                        : Disabled
    Persistence Mode                      : Disabled
    MIG Mode
        Current                           : N/A
        Pending                           : N/A
    Accounting Mode                       : Disabled
    Accounting Mode Buffer Size           : 4000
    Driver Model
        Current                           : N/A
        Pending                           : N/A
    Serial Number                         : 1324419051962
    GPU UUID                              : GPU-ee9b221a-863b-23c4-bd0c-af2c04f3655e
    Minor Number                          : 0
    VBIOS Version                         : 90.02.2E.00.0C
    MultiGPU Board                        : No
    Board ID                              : 0x1800
    GPU Part Number                       : 900-1G150-2500-000
    Module ID                             : 0
    Inforom Version
        Image Version                     : G001.0000.02.04
        OEM Object                        : 1.1
        ECC Object                        : N/A
        Power Management Object           : N/A
    GPU Operation Mode
        Current                           : N/A
        Pending                           : N/A
    GSP Firmware Version                  : N/A
    GPU Virtualization Mode
```

- srun --partition=shpc --gres=gpu:4 clinfo
    - clinfo도 man을 사용하여 의미를 알아보았습니다.



```
CLINFO(1)                  General Commands Manual                 CLINFO(1)

NAME
       clinfo - show OpenCL platforms and devices

SYNOPSIS
       clinfo [options ...]

DESCRIPTION
       clinfo  prints  all  available  information  about all OpenCL platforms
       available on the system and the devices they expose.
```

- 실행 결과

```
Number of platforms                               1
  Platform Name                                   NVIDIA CUDA
  Platform Vendor                                 NVIDIA Corporation
  Platform Version                                OpenCL 3.0 CUDA 11.4.94
  Platform Profile                                FULL_PROFILE
  Platform Extensions
  cl_khr_global_int32_base_atomics cl_khr_global_int32_extended_atomics
  cl_khr_local_int32_base_atomics cl_khr_local_int32_extended_atomics
  cl_khr_fp64 cl_khr_3d_image_writes cl_khr_byte_addressable_store
  cl_khr_icd cl_khr_gl_sharing cl_nv_compiler_options
  cl_nv_device_attribute_query cl_nv_pragma_unroll cl_nv_copy_opts
  cl_nv_create_buffer cl_khr_int64_base_atomics
  cl_khr_int64_extended_atomics cl_khr_device_uuid cl_khr_pci_bus_info
  Platform Host timer resolution                  0ns
  Platform Extensions function suffix             NV

  Platform Name                                   NVIDIA CUDA
Number of devices                                 4
  Device Name                                     NVIDIA TITAN RTX
  Device Vendor                                   NVIDIA Corporation
  Device Vendor ID                                0x10de
  Device Version                                  OpenCL 3.0 CUDA
  Driver Version                                  470.57.02
  Device OpenCL C Version                         OpenCL C 1.2
  Device Type                                     GPU
  Device Topology (NV)                            PCI-E, 18:00.0
  Device Profile                                  FULL_PROFILE
  Device Available                                Yes
  Compiler Available                              Yes
  Linker Available                                Yes
  Max compute units                               72
  Max clock frequency                             1770MHz
  Compute Capability (NV)                         7.5
  Device Partition                                (core)
    Max number of sub-devices                     1
    Supported partition types                     None
  Max work item dimensions                        3
  Max work item sizes                             1024x1024x64
  Max work group size                             1024
  Preferred work group size multiple              32
  Warp size (NV)                                  32
  Max sub-groups per work group                   0
  Preferred / native vector sizes
    char                                                 1 / 1
    short                                                1 / 1
    int                                                  1 / 1
```

(b) GPU 모델명과 노드당 GPU 개수

- 모델명: NVIDIA TITAN RTX
- 노드당 GPU 개수: 4개

(c) GPU하나의 메모리 크기

- 24220 MiB

(d) GPU의 maximum power limit(W)과 maximum SM clock speed

- 파워 리밋: 320.00 W
- 최대 SM 클럭 속도: 2100 MHz

(e) OpenCL을 사용할 때 Max work item dim, Max work item size, Max work group size

- Max work item dimension: 3
- Max work item size: 1024x1024x64
- Max work group size: 1024

2. Matmul with OpenCL

2.1 병렬화 방식에 대한 설명

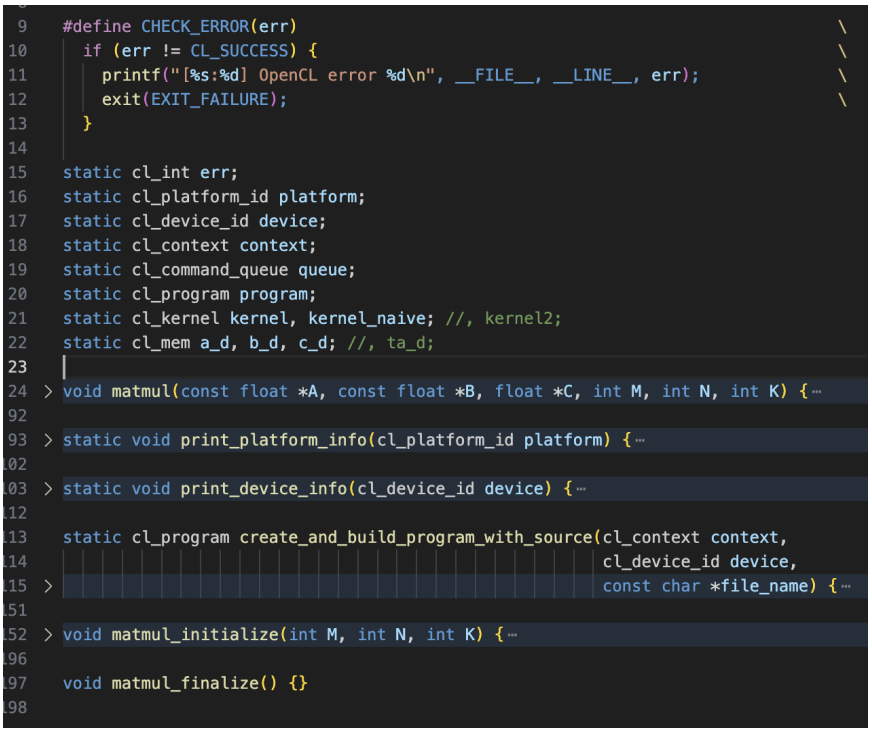
```
__kernel void sgemm(__global float *A, __global float *B, __global float *C, int
M, int N, int K) {
 // TODO: FILL_IN_HERE

 // A: M x K
 // B: K x N
 // C: M x N

 // Ap: K x M
 // Bp: N x K

 // 0 ... col_size
 const int col = get_local_id(0);
 // 0 ... row_size
 const int row = get_local_id(1);
 const int col_size = get_local_size(0);
 const int row_size = get_local_size(1);
 // n - col idx
 const int global_col = TS * get_group_id(0) + col;
 // m - row idx
 const int global_row = TS * get_group_id(1) + row;

 __local float Asub[TS][TS];
 __local float Bsub[TS][TS];
```

```
float c = 0.0;
int t = 0;
const int numTiles = K / TS;
for (int t = 0; t < numTiles; ++t) {
  const int tiledRow = TS * t + row;
  const int tiledCol = TS * t + col;
  Asub[row][col] = A[tiledCol + K * global_row];
  Bsub[row][col] = B[global_col + N * tiledRow];

  barrier(CLK_LOCAL_MEM_FENCE);

  for(int k = 0; k < TS; k++) {
    c += Asub[row][k] * Bsub[k][col];
  }

  barrier(CLK_LOCAL_MEM_FENCE);
}

C[global_col + N * global_row] = c;
}
```

저는 tiling을 통해 속도를 높였습니다. A, B는 transpose하지 않고 그대로 들고 왔으며 tile용으로 Asub, Bsub을 local mem에 만들었습니다. 이후 global memory를 local memory로 로드해왔습니다. 이때 local memory가 한 그룹 안 스레드끼리 공유되는 것을 사용하여 스레드가 각자 부분을 맡아서 속도를 높였습니다. 또 Global memory의 접근 방법이 중요한데, A가 M x K, B가 K x N이므로 N, K, M 순으로 순회하는 것이 좋으므로 A는 k + K * m, B는 n + N * k로 인덱스를 접근했습니다. 이후 k를 순회하면서 sum을 했고 이것을 C에다 저장했습니다.

2.2 뼈대코드의 설명 및 OpenCL API 설명

2.2.1 matmul.c 각 부분에 대한 설명



```
#define CHECK_ERROR(err)                                                \
  if (err != CL_SUCCESS) {                                              \
    printf("[%s:%d] OpenCL error %d\n", __FILE__, __LINE__, err);       \
    exit(EXIT_FAILURE);                                                 \
  }

static cl_int err;
static cl_platform_id platform;
static cl_device_id device;
static cl_context context;
static cl_command_queue queue;
static cl_program program;
static cl_kernel kernel, kernel_naive; //, kernel2;
static cl_mem a_d, b_d, c_d; //, ta_d;

void matmul(const float *A, const float *B, float *C, int M, int N, int K) {…

static void print_platform_info(cl_platform_id platform) {…

static void print_device_info(cl_device_id device) {…

static cl_program create_and_build_program_with_source(cl_context context,
                                                       cl_device_id device,
                                                       const char *file_name) {…

void matmul_initialize(int M, int N, int K) {…

void matmul_finalize() {}
```

main.c 에서는 matmul_initialize, matmul, matmul_finalize만 호출합니다. 이때 matmul_initialize에서 print_platform_info, print_device_info를 호출하여 플랫폼/디바이스 정보들을 출력합니다. 또한 create_and_build_program_with_source함수를 통해 소스 파일을 읽어서 cl_program을 만듭니다.

2.2.2 matmul_initialize

- **clGetPlatformIDs**: 가능한 플랫폼의 리스트를 가져옵니다.
- **clGetPlatformInfo**: 해당하는 플랫폼의 정보를 가져옵니다.
- **clGetDeviceIDs**: 플랫폼에서 가능한 디바이스 리스트를 가져옵니다.
- **clGetDeviceInfo**: OpenCL디바이스의 정보를 가져옵니다.
- **clCreateProgramWithSource**: context를 위해 program object를 만들고 소스코드를 불러옵니다.
- **clBuildProgram**: 소스코드에서 프로그램을 빌드 (컴파일/링킹) 합니다.
- **clGetProgramBuildInfo**: program object에서 각 디바이스에 대한 빌드정보를 가져옵니다.
- **clCreateContext**: OpenCL context를 만듭니다.
- **clCreateCommandQueue**: OpenCL 작업 큐를 만듭니다.
- **clCreateKernel**: kernel object를 만듭니다.
- **clCreateBuffer**: buffer object를 만듭니다.

2.2.3 matmul

- **clSetKernelArg**: 해당하는 kernel의 argument들을 지정합니다.
- **clEnqueueWriteBuffer**: queue에 host memory에 있는 값을 해당하는 버퍼로 적는 작업을 enqueue합니다.
- **clEnqueueNDRangeKernel**: 인자에 맞게 kernel을 실행하는 작업을 enqueue합니다.
- **clEnqueueReadBuffer**: queue에 버퍼에 있는 값을 해당하는 host memory로 읽어오는 작업을 enqueue합니다.
- **clFinish**: 현재 queue에 있는 작업들이 끝날 때까지 대기합니다.

2.2.4 matmul_finalize

코드 없음

2.3 최적화 방식 분류 및 성능 실험 결과

- naive matmul
    - 제출한 코드에서 sgemm_naive 코드를 돌렸을 때, N방향 local_work_size를 32, M방향을 1로 했을 때 1100 GFLOPS 근처로 나왔습니다.
- naive + transpose
    - naive에서 A를 transpose하여서 코드를 돌렸을 때 N, M모두 local_work_size 32일 때 최대 1185 GFLOPS 로 나왔습니다.
- tiling
    - 타일링을 적용하고 나서 1400 GFLOPS 근처로 나왔습니다.
- tiling + transpose
    - matmul의 연산속도는 tiling과 거의 비슷했지만 transpose하는 것에 시간이 상당히 들어가서 오히려 속도가 느려졌습니다.
    - 이것의 경우 transpose를 local memory를 사용하면 조금 더 속도를 높일 수 있는 것으로 보이지만 이번 과제의 경우에는 진행하지 않았습니다.
    -